SHAGE: A Framework for Self-managed Robot Software

Sooyong Park
<sypark@sogang.ac.kr>
Software Systems Engineering Lab.
Sogang University
May. 21~22, 2006

ICSE 2006 Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), Shanghai, China

Outline

• Introduction
• An Approach
• SHAGE Framework
• Demonstration
• Research Issues
• Conclusions
Home Service Robots for the Elderly

- **Home service robots (HSR)** provide useful services such as
  - Face recognition
  - Navigation
  - Bringing an object
  - Conversation
  - ...

- **CIR (the Center for Intelligent Robotics) at KIST (the Korea Institute of Science and Technology)** has been developed various home service robots.
  - T-Rot (High-end), Infotainment robot, H-Robot (Healthcare robot)

---

Overall S/W Architecture\(^1\) of HSR

Research Motivation

Malfunction

User’s new needs

new Environment

Limited Resources

Self-Healing, Adaptive, and Growing Technologies for Intelligent Robots

Example – (1)

- Need for safety: an old man says “move carefully!”
  - During a party: a lot of unrecorded moving objects

**Quality requirement:**
Don’t care execution time, but avoid all possible collisions

Reconfigured Architecture:
ready to use various sensors and to make a map by using the sensors.
Example – (2)

- Need for agility: an old man says “move quickly!”
  - The old man is home alone: no unrecorded object

Quality requirement:
Short response time, Short execution time, optimize use of batteries.

Reconfigured Architecture:
ready to use one sensor (laser).
It cannot avoid (unrecorded) moving objects

Our Approach

- Monitoring
- Brokering
- Decision Making
- Reconfiguration
- Learning
SHAGE Framework

- SHAGE (Self-Healing, Adaptive, and Growing SoftwarE) Framework integrates following technologies
  - Monitoring
  - Brokering: Ontology (authoring relations between environmental information and architectural information)
  - Decision & Learning: Case-Based Decision Theory
  - Reconfiguration: Slot-based architectural style

- This framework provides a test bed for self-managed software.

SHAGE Overall Architecture
A set of Inclination-Situation Pairs
<i1, s1>
<i1, s2>
<i1, sm>

Current Situation (given by the Task Manager)

Candidate Set (reduced by the Architecture Broker)

A set of architecture configurations (Abstract Level Architectures)

Architecture/Component Broker

- Role
  - Searching abstract-level architecture configurations related to the current situation.

- Technology
  - Ontological descriptions.

- Current Status
  - It can only search in a small set of configurations related to the navigation subsystem.
  - Rule-based search: it cannot relax rules.
Decision Maker & Learner

- **Role**
  - **Selecting** exactly one configuration and one component for each slot from the candidate set retrieved by the architecture broker.

- **Technology**
  - Case-Based Decision Theory

- **Current Status**
  - It only carries out in limited scope.
  - Limited search space: only in the navigation subsystem.
  - Limited learning time: few scenarios.
Reconfigurator

- **Role**
  - Reconfiguring the current software architecture dynamically.

- **Technology**
  - Slot-based two-level software architectural style.

- **Current Status**
  - It has reconfigured only the navigation subsystem.
    - All configurations for the subsystem were verified in the demonstration.
  - It can manage components distributed in SBCs (Single Board Computers) by RMI.
  - It supports components implemented in Java and C++ (through JNI).
Demonstration

Research Issues

- Internal monitoring
- Ontology construction
- Learning speed
- Run-time measurement and validation
- Componentization
- Domain Knowledge
Conclusions

• **SHAGE Framework** has been developed to provide 'self-managing capabilities' to robot software.

• The framework integrated ontology, decision theory, and dynamic architecture and comprises
  – Monitor
  – Architecture/Component Broker
  – Decision Maker & Learner
  – Reconfigurator

• In the experiment, we showed a simple scenario.
  – The robot managed its architecture to adapt its changing requirements.

• The second phase (three years) of our research just launched to improve key technologies.